Algoritmos y Estructuras de Datos III - 2$^{\underline{\text{do}}}$ Recuperatorio
Fecha de examen: 24-JUN-2022

	Nº Orden	Apellido y nombre		L.U.	# hojas [1]
Notas:	Ej1	Ej2	Ej3	Ej4	Final

Aclaraciones: El parcial se aprueba con 2 (dos) ejercicios aprobados. Cada hoja debe estar numerada y debe tener el número de orden y L.U. El parcial dura 4 horas y es a libro abierto. Cada respuesta dada debe contar con su correspondiente **justificación** para poder ser considerada correcta.

1) Para resolver el Ejercicio 1 del segundo parcial de este cuatrimestre, bastaba con ejecutar Dijkstra sobre un digrafo fuertemente conexo cuyas aristas tienen peso 0 o 1. Según Santiago, Dijkstra se puede implementar en tiempo lineal en este tipo de digrafos. El objetivo de este ejercicio es probar la afirmación de Santiago, para lo cual podemos utilizar la siguiente descripción del algoritmo de Dijkstra que fue presentada en alguna teórica:

   Algoritmo de Dijkstra $0-1$
   Input: un digrafo D fuertemente conexo tal que cada arista $v \to w$ tiene peso $p(vw) \in \{0,1\}$ y un vértice inicial s.
   Output: vector δ de distancias de s a todos los vértices de D.

   1. Inicializar un conjunto $Q = \{(s,0)\}$ y poner $\delta(v) = \infty$ para todo $v \in V(D)$.
   2. Mientras $Q \neq \emptyset$:
   3.   Sacar de Q el par (v,d) con mínimo d.
   4.   Si $\delta(v) = \infty$:
   5.   Poner $\delta(v) = d$.
   6.   Insertar en Q el par $(w, d + p(v \to w))$ para todo $w \in N^{\text{out}}(v)$.

   a) Determinar un invariante que satisfagan los pares (v,d) que se encuentran en Q en cada iteración del algoritmo de Dijkstra tal que nos ayude a implementar el algoritmo en tiempo lineal.

   b) Describir la implementación del algoritmo utilizando la siguiente ayuda de Santiago: "Ya sabemos que Dijkstra se puede implementar usando una cola de prioridad en la que cada arista se inserta $O(1)$ veces (ver Teórica). Por lo tanto, alcanza con diseñar una cola de prioridad que, aprovechando el invariante, permita encolar y desencolar en $O(1)$ tiempo".

2) Tenemos a n clientes de un supermercado $\{c_1, c_2, \ldots, c_n\}$ y queremos asignarle a cada uno una caja para hacer fila. Las cajas están ordenadas en forma circular, numeradas de la 1 a la M y se encuentran separadas por pasillos. Entre la caja M y la 1 hay una valla que impide pasar de una a la otra. Durante el proceso de asignación algunos clientes se pelean entre sí y son separados por seguridad. Si dos clientes c_i y c_j se pelean, los guardias les dicen que tienen que ponerse en filas distintas que se encuentren separadas por al menos $K_{ij} > 0$ pasillos intermedios en ambos sentidos del círculo, para que no se vuelvan a pelear. Notar que cuando seguridad separa una pelea naturalmente hay un cliente que queda en un número de caja más bajo y el otro en un número de caja más alto. Con la restricción de no volver a acercarse y la valla entre las cajas M y 1 ese orden ya no puede cambiar. ¿Será posible asignarlos a todos?

   a) Modelar el problema utilizando un sistema de restricciones de diferencias (no olviden justificar). Para el modelo, notar que sabemos qué clientes se pelearon. Más aún, si c_i y c_j se pelearon, sabemos quién entre c_i y c_j quedó del lado de las cajas con menor numeración.

[1] Incluyendo esta hoja.

b) Proponer un algoritmo polinomial que lo resuelva.

c) ¿Qué complejidad tiene el algoritmo propuesto? Para la respuesta, tener en cuenta la cantidad m_1 de peleas.

3) Una 2-CDC (combination of difference constraints) es una secuencia $C_1, \ldots, C_k$ de cláusulas donde cada cláusula C_i puede ser:

- Una ecuación del tipo $x_i = 0$
- Una desigualdad de la forma $x_i - x_j \leq c$
- La disyunción de dos desigualdades de la forma $(x_i - x_j \leq c) \vee (x_k - x_l \leq c)$.

donde c es una constante entera, mientras que las x_i son variables de $\mathbb{Z}$. Decimos que un 2-CDC es satisfacible si existe una asignación de valores enteros a las variables tal que cada cláusula es verdadera. Por ejemplo: $C_1 : (x_1 = 0), C_2 : ((x_1 - x_2 \leq 3) \vee (x_3 - x_1 \leq 2)), C_3 : (x_3 - x_2 \leq 8)$ se satisface con $x_1 = 0, x_2 = -3, x_3 = 5$, mientras que $C_1 : (x_1 = 0), C_2 : (x_2 = 0), C_3 : (x_1 - x_2 \leq -1) \vee (x_2 - x_1 \leq -1)$ no es satisfacible.

Demostrar que el problema de determinar si un 2-CDC es satisfacible es NP-completo sabiendo que 3-coloreo[2] es NP-completo.

Ayuda: piense en codificar cada vértice v del grafo usando una variable x_v en el dominio $[0, 2]$ de forma tal que x_v represente el color que se le asigna a v en un 3-coloreo.

4) En un hospital hay k períodos disjuntos de vacaciones (por ej. Semana Santa, Feriado puente de Güemes, etc.). Cada período consiste de una cierta cantidad de días contiguos de vacaciones. Conocemos cada conjunto $D_j = \{d_{j1} \ldots, d_{jr}\}$ de días contiguos que conforman el período j. En este hospital hay n medicxs y cada cual tiene un conjunto S_i de días disponibles para trabajar durante los períodos de vacaciones. Por ejemplo, una médica puede tener disponible viernes y sábado de semana santa y el lunes del feriado de Güemes. Queremos encontrar una asignación que cumpla:

- Nadie tiene asignado más que C días totales para trabajar en vacaciones (y solo dentro de sus días disponibles).
- Cada día de vacaciones tiene asignada una única persona para trabajar ese día.
- Para cada período j unx medicx sólo puede tener, como máximo, un día asignado dentro de D_j. Es decir, quizá tiene disponibles jueves, viernes, sábado y domingo de semana santa pero solo se le puede asignar uno de esos días.

a) Modelar el problema como un problema de flujo máximo (no olviden justificar).

b) Proponer un algoritmo polinomial que resuelva el modelo propuesto.

c) ¿Qué complejidad tiene el algoritmo propuesto?

Ayuda: Consideren la posibilidad de agregar nodos intermedios que no representen ni a lxs médicxs ni a los días de vacaciones pero que sirvan para cumplir las restricciones.

[2]Recordar que dado un grafo G, 3-coloreo consiste en determinar si se puede particionar a los vértices de G en a lo sumo 3 conjuntos independientes.